Project Report: Calendar Sync Cloud Application

Introduction

This project is a Calendar Sync Cloud Application that enables users to synchronize their calendar events across multiple devices. Built using Flask and deployed on AWS, it offers real-time event management with seamless cloud integration using AWS DynamoDB, S3 bucket storage, and Lambda functions.

Objectives

- Develop a web-based calendar application using Flask.
- Implement event management functionalities (add, update, delete).
- Enable real-time data synchronization using AWS.
- Provide a responsive and user-friendly interface.
- Utilize AWS services for data storage, compute, and serverless management.

Technologies Used

- **Python**: For backend logic and server-side operations.
- **Flask**: For building the web framework.
- **AWS DynamoDB**: For cloud-based storage of calendar events.
- **AWS S3**: For storing application data and static files.
- **AWS Lambda**: For serverless functions to handle CRUD operations.
- **HTML, CSS, JavaScript**: For the front-end interface.

Features

1. **Event Management**: Users can add, edit, and delete events.
2. **Cloud Syncing**: Events are stored in AWS DynamoDB, ensuring data is accessible from multiple devices.

3. **File Management**: S3 buckets store additional data like images or attachments for events.
4. **Serverless Functions**: AWS Lambda ensures efficient backend operations without managing servers.
5. **Responsive UI**: Built with HTML, CSS, and JavaScript for a seamless user experience.

Project Architecture

- **Frontend**: HTML forms for event input, displayed using JavaScript.
- **Backend**: Flask handles API requests for CRUD operations.
- **Database**: AWS DynamoDB stores event data.
- **File Storage**: AWS S3 bucket stores relevant files like images or attachments.
- **Compute**: AWS Lambda functions process requests without dedicated server management.

API Endpoints

- `POST /add_event`: Adds a new event.
- `GET /events`: Retrieves all events from DynamoDB.
- `PUT /update_event`: Updates an event.
- `DELETE /delete_event`: Deletes an event.

Future Enhancements

- Implement authentication using AWS Cognito.
- Add calendar views for better visualization.
- Enable notifications and reminders for events.
- Expand data storage options with AWS RDS.

Conclusion

The Calendar Sync Cloud Application offers an efficient solution for managing events across multiple devices. Its integration with AWS using DynamoDB, S3, and Lambda ensures secure and reliable data synchronization, making it a robust choice for personal or business use.

References

- Flask Documentation
- AWS Documentation
- DynamoDB Documentation
- S3 Documentation
- AWS Lambda Documentation

Project Developed By: Soumya Pachal